REMARKS

This responds to the Office Action mailed on June 17, 2004.

No claims are amended, no claims are canceled, and no claims are added; as a result, claims 1-21 are now pending in this application.

§102 Rejection of the Claims

Claims 1-18 were rejected under 35 USC § 102(e) as being anticipated by Muller et al. (U.S. 6,453,360).

Regarding the use of the valid bit in embodiments of the present method and apparatus, the present specification teaches the following.

The send queue packet sequence number tracking circular queue in Fig. 8 tracks each read request packet sequence number and the last transmitted packet sequence number so it can validate response packets. For every transmitted packet, the packet sequence number is written to the location pointed to by write pointer 802 and the valid bit 803 is set. If the packet is a read request packet, the read indicator is set and write pointer 802 is incremented so the information will not be overwritten. For all other packet types, write pointer 802 is not incremented and the read indicator is cleared. This will allow the next write to the circular queue to overwrite the current packet sequence number, indicating that the previous packet may be implicitly acknowledged.

When a response packet is received, the packet sequence number is checked against the packet sequence number stored in the location in the circular queue pointed to by read pointer 801. If valid bit 803 for this entry is not set, no responses are expected and the packet is dropped. If valid bit 803 for this entry is set, the response packet sequence number must be equal to or earlier than the entry in the circular queue and after the last acknowledged packet sequence number. If the response packet is a read response, the packet sequence number must match the entry in the circular queue and the entry must be marked read. If the response packet sequence number matches the entry, the entry pointed to by read pointer 801 has valid bit 803 cleared and read pointer 801 is incremented after the packet has completed.

The read queue packet sequence number tracking circular queue in Fig. 9 tracks each read request packet sequence number and the last received packet sequence number so it can transmit response packets. For every received packet, the packet sequence number is written to the location pointed to by write pointer 902 and valid bit 903 is set. If the packet is a read request packet, read bit 904 is set and write pointer 902 is incremented so the information will not be overwritten. For all other packet types, write pointer 902 is not incremented and the read indicator is cleared. This will allow the next write to the circular queue to overwrite the current packet sequence number and allow the packet to be implicitly acknowledged. When a response packet is to be transmitted, the packet sequence number pointed to by read pointer 901 is read. If the valid bit at the location is not set, nothing is transmitted. If read bit 904 is set, a read response packet is transmitted. Otherwise, an acknowledge response is transmitted. Once the packet has completed, valid bit 903 is cleared and read pointer 901 is incremented.

The Examiner has alleged that Muller shows an operational code which functions to provide information to DMA engine to assist in the re-assembly of the data packet. The Examiner further states that important flags are set to indicate more data is likely to come or not. For example, flags are checked to indicate if more data is to follow (see fig. 6B, 618-626) and operational codes indicate flows that are expected to follow (col. 44 lines 52-53 and col. 45 lines 26-35).

However, Muller et al teach that in state 636, operation code 7 is selected for the packet. In the present context, operation code 7 indicates that the packet is compatible, matches a valid flow and contains data. Operation code 7 may further signify, in this context, that the packet constitutes an attempt to re-synchronize or reset a communication flow/connection and that additional data is expected once the flow/connection is reset. In effect, therefore, the existing flow is torn down and a new one (with the same flow key) is stored in its place. After state 636, the illustrated procedure ends at end state 670.

Specifically, claims 1 and 10 have include "<u>wherein the valid bit is indicative of whether at least one response is expected</u>" (see page 20, line 5 of the application). This feature is neither shown nor suggested in Muller. Muller does not apply the "valid bit" as set forth in the amended claims. Rather, Muller teaches using the valid bit for error determination or correction, or as a validity indicator.

Claims 19-21 were rejected under 35 USC § 102(e) as being anticipated by Dobecki (U.S. 6,611,879).

However, Dobecki teaches that (see FIG. 7), the exemplary director 180.sub.1 is shown to include in the message engine/CPU controller 314. The message engine/CPU controller 314 is contained in a field programmable gate array (FPGA). The message engine (ME) 315 is coupled to the CPU bus 317 and the DMA section 408 as shown. The message engine (ME) 315 includes a Direct Memory Access (DMA) section 408, a message engine (ME) state machine 410, a transmit buffer 424 and receive buffer 424, a MAC packetizer/depacketizer 428, send and receive pointer registers 420, and a parity generator 321. The DMA section 408 includes a DMA transmitter 418, shown and to be described below in detail in connection with FIG. 9, and a DMA receiver 424, shown and to be described below in detail in connection with FIG. 10, each of which is coupled to the CPU bus interface 317, as shown in FIG. 7. The message engine (ME) 315 includes a transmit data buffer 422 coupled to the DMA transmitter 418, a receive data buffer 424 coupled to the DMA receiver 421, registers 420 coupled to the CPU bus 317 through an address decoder 401, the packetizer/de-packetizer 428, described above, coupled to the transmit data buffer 422, the receive data buffer 424 and the crossbar switch 320, as shown, and a parity generator 321 coupled between the transmit data buffer 422 and the crossbar switch 320. More particularly, the packetizer portion 428P is used to packetize the message payload into a MAC packet (FIG. 2B) passing from the transmit data buffer 422 to the crossbar switch 320 and the de-packetizer portion 428D is used to de-packetize the MAC packet into message payload data passing from the crossbar switch 320 to the receive data buffer 424. The packetization is here performed by a MAC core which builds a MAC packet and appends to each message such things as a source and destination address designation indicating the director sending and receiving the message and a cyclic redundancy check (CRC), as described above. The message engine (ME) 315 also includes: a receive write pointer 450, a receive read pointer 452, a send write pointer 454, and a send read pointer 456.

Dubecki also explains that all four pointers 450, 452, 454 and 456 are reset to zero on power-up. As is also noted above, the message engine/CPU controller 314 also includes: the de-packetizer portion 428D of packetizer/de-packetizer 428, coupled to the receive data buffer 424

(FIG. 7) and a packetizer portion 428P of the packetizer/de-packetizer 428, coupled to the transmit data buffer 422 (FIG. 7).

Thus, it is clear that the packetizer portion 428P and the de-packetizer portion 428D are contained in a single packetizer/de-packetizer (a single engine) 428, and are not separate as disclosed in the present specification. In fact the Examiner has stated "that Dobecki shows a message engine further comprising of packetizer and depacektizer". Thus, Dobecki teaches one message engine, whereas Applicant claims a receive queue engine partitioned from a send queue engine (that is, two engines).

Claim 19 sets forth "a receive queue engine partitioned from the send queue engine". Support for this amendment is included on pages 15 and 16 in the description of Figure 6 of the application. The features of the claim 19 are not shown in Dobecki which uses a single packetizer/de-packetizer 428.

Conclusion

For the reasons set forth above the Examiner is respectfully requested to reconsider the rejection of the claims. Applicant respectfully submits that the claims are in condition for allowance and notification to that effect is earnestly requested. The Examiner is invited to telephone Applicant's attorney John Garrett at 847-740-9080, or the below-signed attorney at (612) 349-9592, to facilitate prosecution of this application.

If necessary, please charge any additional fees or credit overpayment to Deposit Account No. 19-0743.

Respectfully submitted,

BRIAN M. LEITNER ET AL.

By their Representatives,

SCHWEGMAN, LUNDBERG, WOESSNER & KLUTH, P.A.
Attorneys for Intel Corporation
P.O. Box 2938
Minneapolis, Minnesota 55402
(612) 349-9592

Date Aug. 17, 2004 By Ann M. McCrackin
Ann M. McCrackin
Reg. No. 42,858

CERTIFICATE UNDER 37 CFR 1.8: The undersigned hereby certifies that this correspondence is being deposited with the United States Postal Service with sufficient postage as first class mail, in an envelope addressed to: Mail Stop AF, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450, on this 17TH day of August, 2004.

Chris Hammond | Chris Hammond

Name | Signature